REMARKS

Claims 1-16 are pending. Claims 1, 7 and 12 are amended. Claims 2, 8 and 13 are canceled. The remaining claims are unchanged.

In the Office Action, claims 1-16 were rejected under 35 U.S.C. § 103(a) as obvious in view of Long (U.S. Patent No. 6,691,307) and Seshadri (U.S. Patent No. 6,658,421).

The rejection of claims 1-16 should be withdrawn for the reasons below.

Claim 1, as newly amended to include the subject matter of claim 2 as filed, recites a feature which the cited art fails to disclose or suggest, namely:

> A computer system comprising:
>
> a preloader arranged to,
>
> determine whether a bytecode makes an active reference to a class which requires an execution of a static initializer,
>
> determine if the class has a superclass which requires the execution of the static initializer, wherein the preloader produces a source file,
>
> rewrite the bytecode to a new bytecode which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which requires the execution of the static initializer; . . ..
>
> (Emphasis Added).

The computer system defined by claim 1 includes a preloader which provides several features. The preloader is arranged to make the determinations recited above, and to "rewrite the bytecode to a new bytecode which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which requires the execution of the static initializer." In this way, in one embodiment, the preloader can reduce the number of runtime checks for static initializers during execution of the object file.

The Office Action states, on page 5, lines 1-6, that Long teaches a preloader arranged to rewrite the bytecode to a new bytecode in the manner specified by newly amended claim 1. The Office Action states that the above-quoted feature of claim 1 is disclosed in the form of preloader 172 and runtime system 174 (col. 7:22-33, Fig. 9A). Applicant disagrees with this statement.

Long fails to disclose or suggest a preloader arranged in the manner recited in claim 1. Instead, Long teaches a conventional preloader: "[t]he preloader 172 loads a classfile 170 and organizes code within the classfile according to native endianness ... and outputs code and data structures for the runtime system 174." (col. 7, lines 24-28). While Long suggests organizing code according to native endianness, Long fails to make any mention of rewriting a bytecode to a new bytecode which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which requires the execution of the static initializer, as recited in claim 1.

The Long patent is concerned with interpreter optimization for native endianness (col. 4, lines 9-10), not eliminating unnecessary runtime checks for static initializers. Accordingly, Long fails to disclose or suggest making any determination whether the bytecode makes an active reference to a class which requires execution of a static initializer, or if the class has a superclass which requires the execution of the static initializer, as stated in the Office Action at page 4, lines 7-10. Without making such a determination, those skilled in the art would not have been encouraged to "rewrite the bytecode to a new bytecode which indicates that at least one of the class and the superclass requires execution of the static initializer," as recited in claim 1. Long fails to disclose or suggest any such feature. Consequently, Long fails to anticipate or render obvious the computer system defined by claim 1.

The Office Action further states, on pages 2-3, that "Seshadri in an analogous art teaches 'a preloader arranged to, determine whether a bytecode (E.g. see col. 4:56, invokestatic) makes an active reference to a class which requires an execution of a static initializer, determine if the

class has a superclass (E.g. see col. 4:66 and col. 5:5) which requires the execution of the static initializer, wherein the preloader produces a source file.' (E.g. see TABLE 2 at col. 12 and see col. 4:54 to col. 5:5)."

For the sake of argument, even if the above-cited portions of Seshadri support these assertions, Seshadri fails to disclose or suggest "rewrit[ing] the bytecode to a new bytecode which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which requires the execution of the static initializer," as recited in claim 1. Accordingly, Seshadri would fail to cure the deficiencies of Long even if Seshadri could somehow be combined with Long.

Seshadri discusses replacing explicit names in the bytecodes with machine addresses or hardwired offsets (col. 13, lines 20-22), and encoding a signature when compiling an invokestatic bytecode. (col. 4, lines 55-57). Nowhere, however, does Seshadri suggest rewriting the bytecode to a new bytecode which indicates that at least one of the class and the superclass requires execution of the static initializer when it is determined that the bytecode makes the active reference to the class which requires the execution of the static initializer, as recited in claim 1. The generating and encoding of signatures described by Seshadri is for detecting binary incompatibility in object code, (col. 4, lines 9-21), not eliminating unnecessary runtime checks for static initializers. Accordingly, Seshadri is not concerned with rewriting a bytecode to a new bytecode which indicates that at least one of a class and a superclass requires execution of a static initializer, and fails to offer any teaching in this regard.

Because Seshadri fails to disclose or suggest the above-quoted features of the computer system of claim 1, Seshadri fails to support an obviousness rejection of claim 1 under 35 U.S.C. § 103(a), taken alone or in combination with Long. Therefore, this rejection should be withdrawn.

Independent claims 6, 7 and 12 incorporate similar features as claim 1. Thus, the cited art, taken alone or in combination, fails to support a rejection of these claims for the same reasons as claim 1.

Dependent claims 3-5, 9-11 and 14-16 incorporate all of the features of the independent claims on which they are based. Therefore, the cited art fails to support any rejection of these claims for the same reasons as the independent claims on which they are based.

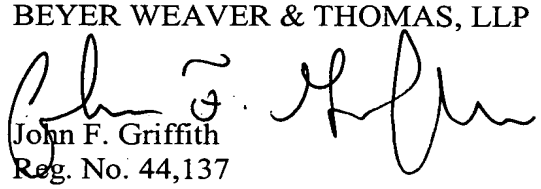
CONCLUSION

Based on the foregoing, it is respectfully submitted that this application is in condition for allowance. Early notification to that effect is respectfully requested.

If there are any issues remaining after the review of this Response, the Examiner is respectfully requested to contact the undersigned at the telephone number below.

If any fees are due in connection with the filing of this Response (including any fees due for an extension of time), such fees may be charged to Deposit Account No. 500388 (Order No. SUN1P802).

Respectfully submitted,
BEYER WEAVER & THOMAS, LLP



John F. Griffith
Reg. No. 44,137

P.O. Box 70250
Oakland, CA 94612-0250